22. (*expires 4/12*) Historically, "pen and paper" encryption (that is, what was mostly done prior to about 1980) would first rewrite a message to be encrypted in all upper-case letters, omitting all punctuation and spaces, and then writing out the message in blocks of five letters. For example, the message `Mr. Watson, come here - I want to see you!` would be transcribed as `MRWAT SONCO MEHER EIWAN TTOSE EYOU` prior to encrypting. If the message was sufficiently long (longer than 50 characters or so), the blocks of letters would be continued on a new line.

    Write a pair of Maple procedures to accomplish this goal (in slightly more general terms).

    The first should take as input a string, convert all lower-case letters to upper case, and afterwards, remove any characters that are not in a global variable called `Alphabet`. For this, I recommend using `UpperCase` and `Select` from the `StringTools` package.

    The second procedure should print this out in a format as described above (blocks of 5 letters, separated by spaces). I recommend using `printf` for this.

    As usual, demonstrate that your routines work with some appropriate test cases, with at least one short message as above, and another message of perhaps 100 characters or so. Note that your routine should work with a variety of alphabets.

23. (*expires 4/12*) When we implemented RSA in class, we represented our encrypted messages as a list of large numbers, rather than converting them to printable text. Sometimes we want a text representation. One way to do this to use a base-64 representation, where the message m is converted to a base 64 number. This base 64 number is commonly represented with the upper-case characters `A`–`Z` representing digits 0 through 25, lower-case `a`–`z` representing digits 26 through 51, the characters `0`–`9` representing 52 through 61, and `+` and `/` representing 62 and 63, respectively.

    Write a generalized implementation of this conversion process. Specifically, assume there is a global called `AlphabetOut` which contains the allowed characters in the encoding, ordered appropriately. Your procedure should take as input two arguments: a list of numbers in base n, and the base n. Your procedure should return a string representing the message in base b, where b is the length of `AlphabetOut`. Also write another procedure which undoes this conversion.

    As an example[1], the following list of numbers represents some text converted from ASCII (base 256) to base 10^{47} (that is, without encryption):

    66690138583950401502911241221419634561895711 37, 79085785195062207278272120198122975492318184082, 15624867350544934942834543866565863795868490456, 55387683611779270304689525842891535523935500393, 23896957611465431133603100420167106476881540779, 78091587231828327640146863695263953922927490912, 97646064248467841327319151666448832697087427 61, 24147181471923289394456210178528341598920602555, 59677642432524170063171614096094265077340147036, 10170835876508997111331 2874866

    When transcribed into the base 64 encoding described above, we get:

    ```
    BBCbv52ZgQXatVGIhd2bsASauBSYgcWYsFGe5BiZhJHLgYWYyBSY3FWeu4iLKoQS
    0BSazBSYgAXZyl2bkBybmByYpZXasBydhJnLgIVZiVGbgowcwF2YlNHapB3csAyc
    0JXarlmbnBiZy9WbgEGIolGZkVmbgogYhNXZsACahZXZgc3buBCdoVWayBiZpJ3c
    0BidpNGdvJXegoQYnFWauNHdgQHalBSZ2lGbgcUYsF2Y0l2YgUUbwlmcl5C
    ```

    By the way, this text is from the opening to a well-known movie.

[1] Using Maple's convention of least-significant digit first, so the decimal number 123 is [59,1] (or 7B) in base 64.